Weekly Updates – Vinayak Gajendra Panchal

PART-1: LaneLet2 important takeaways

1. Opensource graph-based map framework for Highly Automated Maps and traffic simulations. Its an upgraded version of Liblanelet.
2. Automated Driving and Mapping Needs: vital role in accurate and comprehensive maps for complex urban scenraios. (need of presice vehicle positions and decision making).
3. HAD maps need: info about vehicle surroundings and overcome sensor limitations.
   - Lane-level accurate maps.
   - Information about sidewalks bikes, tram, and pedestrian lanes (lanelet2 incorporates it).
4. Why lanelet2:
   - OSM data quality: vary a lot as created by volunteers, can be editable (commonly used).
   - No libraries for public by commercial maps providers.
   - OpenDRIVE framework: lacks freely available tools for interpretation and processing.
   - Liblanelet Framework: focuses on motion planning. It represents roads using small lane sections (Lanelets) and includes traffic rules. However, it has limitations, such as only allowing lane changes at predefined points.
   - Lack of suitable, standardized map formats for automated driving.
5. LaneLet2 has 3 layers: each layer connects to the other as graphs. Below is the image taken from the paper which shows the connectivity of layers in certain scenarios.

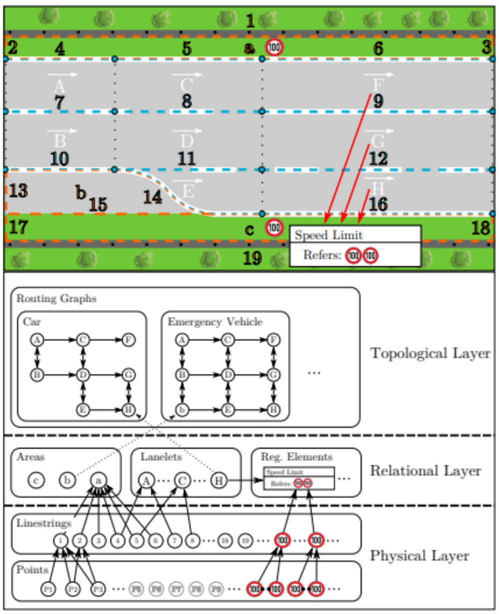

Fig. 2: Map example for a highway road (enclosed by guardrails) and the resulting map structure. Lanelets have capital letters, areas lowercase letters and linestrings have numbers.

6. Every element in the map has a unique ID for easy identification.

7. Attributes can be assigned to elements in the form of key-value pairs, allowing for additional information to be stored.
8. The map ensures that all information is verifiable while driving, by associating it with observable objects.
9. One lanelet can overlap another but they are divided by regulatory elements (traffic rules)
10. **Lanelets and areas** can have various interactions and relationships, making the map dynamic and adaptable to different situations and road users.
11. **Regulatory elements** in Lanelet2 serve as the backbone for implementing traffic rules within the map, guiding automated vehicles through a complex network of legal and practical road usage guidelines.
12. LaneLet2 Modules (High-level)
    - Core: Holds basic map elements and tools for calculating distances and overlaps.
    - Traffic Rules: Interprets road rules based on vehicle type and country, ensuring compliance. (lane change allowed or lanelet permissions)
    - Physical: Provides easy access to physical map elements such as road markings.
    - Routing: Finds the optimal path and potential maneuvers, considering traffic rules.
    - Matching: This module is used to assign lanelets to road users or to determine possible positions on the map based on specific observations of the sensors
    - Projection: Converts global coordinates to local, metric coordinates for calculations.
    - IO (Input/Output): Handles reading and writing of map data, particularly in OSM format.
    - Validity: Checks the map for errors to ensure accuracy and reliability.
    - ROS: Connects Lanelet2 to the Robot Operating System for broader application in robotics.
    - Python: Allows for the use of Lanelet2 functionalities in Python programming language.

PART-2: Models/Ideas to predict/classify collisions at position/lane

1. First, I created a dataset based on Floating Car Data and Collision Data.
2. This dataset is not suitable for classification task as it is an unbalanced dataset with only 18 collision instances (collider and victim) in 40K data points.
3. Since it is a time series data, so as to predict the collision at particular timeframe, we need to use timeseries ML/DL model we can go with Sequence/time-series models like
    - RNN, LSTM, Gated Recurrent Unit (GRU).
    - Probabilistic models like Hidden Markov Models (HMMs)
4. What can we do with this dataset (tweaking the dataset will be required):
    - Collision Prediction: Predict the likelihood of a collision occurring based on attributes like speed, acceleration, distance, and type of vehicle.

- Anomaly Detection: Identify unusual patterns or outliers in the traffic flow, which could indicate accidents or other irregular events.
- Type Classification: Classify the type of vehicle based on attributes like speed, acceleration, and lane.
- Lane Prediction: Predict the lane a vehicle will be in based on its current position, speed, and angle.

5. In a lanelet-based Markov Decision Process, the vehicle's potential locations and directions are represented as states on a lanelet map (for that we have to convert it into lanelet graph format).
    a. States: Each possible position and direction of the car on the map.
    b. Actions: Different driving moves the car can make (go straight, turn, change lanes).
    c. Transitions: Chances of successfully making these moves, considering the car might not always do exactly what we want.
    d. Rewards/Costs: Points given for safe and quick driving, or penalties for unsafe actions or delays.